Electronic Notes in Theoretical Computer Science 69 (2003)
URL: http://www.elsevier.nl/locate/entcs/volume69.html 16 pages

# Paths and Simulations

Krzysztof Worytkiewicz

*LTI, Swiss Federal Institute of Technology, CH1015-Lausanne*

**Abstract**

We study a notion of *path simulation* among *categorical transition systems*, a generalized version of labeled transition systems. We then give a characterization in terms of open maps and, in the relevant case where the labels are spans of sets, the relationship to simulations among corresponding *categories of evolutions.* More algebraic aspects are investigated in a bicategorical setting where path simulations are characterized as *binary predicates over* cts's, living in a bicategory of cylinders. The latter plays the rôle of a *relational structure* in this setting.

## 1 Introduction

This paper is devoted to a study of a notion of simulation called *path simulation* taking place among *categorical transition systems*. Cts's are pseudofunctors given by graph morphisms from graphs expressing *control* into a bicategory of labels expressing (typed) computation, so in particular they are more general than labelled transition systems in that the labels are composable and may moreover exhibit more structure than in the classical setting (*cf.* [18]). Path simulation is a generalization of both strong and weak simulation. Although it is formally a strong simulation, an individual transition may be matched by a proper path (as long as the labels agree), which is similar to but more general as the relationship of weak simulation to the *saturation monad.*

The technique of open maps is fairly standard by now although mainly used to characterize *bisimulation* directly as a span of such, *i.e.* without decomposing into "inverse" simulations (*cf.* [2]). Since we study *simulation* in this paper, the characterization involves a span with only one leg open, not unlike (a weakened version of) the original notion of simulation for presheaves (*cf.* [1]). A relevant special case of cts's are those labelled with spans since imperative programs with communication can be modeled with such a device (*cf.* [8] and also [21]). In this setting, there is an associated *ulf* functor to the free category over the control graph. The construction has the computational reading of a *category of evolutions* w.r.t. the data values of types given by the labeling spans (*cf.* [10]) and we show how path simulation *lifts* from typed computation along control paths to the corresponding categories of evolutions.

We investigate further algebraic aspects of path simulation using techniques derived from those originally considered by Hermida (*cf.* [13]) for the study of simulations among transition systems and provide a characterization of path simulation in terms of *premodules*. In this work, a premodule is an object in a bicategory equipped with a *category action* in form of a lax functor. Path simulation is characterized as a binary predicate over the cts's involved in the simulation, this in the setting of a *bicategory of cylinders*.

The structure of the paper is as follows. Section 2 introduces the notions of *categorical transition systems* and of associated *categories of evolutions*. Section 3 introduces the notion of *path simulation* and contains a study thereof in terms of an appropriate notion of open maps. Section 4 investigates path simulation in terms of *premodules*. Section 5 concludes comparing the material with other approaches and suggesting possibilities how to push further the ideas laid down in the paper.

## 2 Categorical Transition Systems and Evolutions

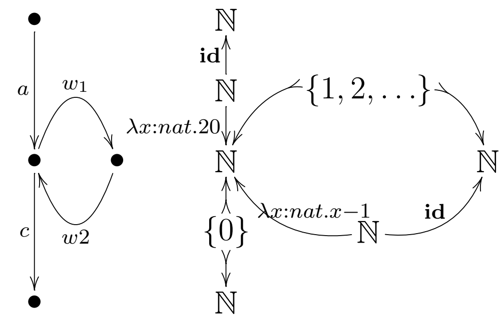
A popular presentation of labeled transition systems consists of set-indexed families oF relations, which boils down to consider certain graph homomorphisms from graphs of transitions to one-vertex graphs with individual labels as self-loops. The homomorphisms need to satisfy a local injectivity condition saying roughly that there are no parallel edges with the same label. In the present section we extend this view to *categorical transition systems* or *cts*'s: the local injectivity is dropped and the labels are organized in a bicategory of spans $\mathbf{Span}\mathbb{B}$ over a category $\mathbb{B}$ (*cf.* [4]). Since it is the case that $\mathbb{B} = \mathbf{Sets}$ in the leading examples, let $\mathbf{Span} \stackrel{def}{=} \mathbf{SpanSets}$.

Consider the program in-context

$$\texttt{x:=20; while x>0 do x:=x-1 end}[x : nat]$$

written in a block-structured imperative language. Let $\mathsf{F} \vdash \mathsf{U} : \mathbf{RGraph} \to \mathbf{Cat}$ be the adjunction from the category $\mathbf{RGraph}$ of *reflexive* graphs[1] to $\mathbf{Cat}$. The above program gives rise to a graph $G$ representing the program's *control flow* and to a pseudo-functor $p : \mathsf{F}G \to \mathbf{Span}$, the graph part of the assignment being suggested by

[1] All graphs in this setup are understood as *reflexive*, *i.e.* equipped with a distinguished self-loop at every vertex.




In particular, the program's *locations* are one-to-one with $G$'s vertices and each transition carries a span reflecting the assignments `x:=20` resp. `x:=x-1`, the evaluation of the branching condition `x>0` or the exit of the `while`-block. Obviously, $p$ is a pseudo-functor since its values on compound paths are given by compositions of spans.

It can be shown that such imperative programs[2] give rise to pseudo-functors as above (*cf.* [21]). Off course, this example does not elucidate the advantage of taking general spans instead of simpler relations nor the reason for considering reflexive graphs. Nonetheless, we will see that reflexive graphs and general spans are essential (in this context) for an account of interaction.

Notice that we could dispense with displaying $G$ in the figure above since it can be deduced from the labeling part and it suffices indeed to display this latter aspect only, provided some conventions are met. Specifically, in order to always be able to deduce the control part we duplicate labels when necessary, similarly to a convention met in the literature discussing sketches (*cf.* [3]). Moreover, we omit to display unit spans coming from $G$'s units.

**Definition 2.1** *Let $\mathcal{K}$ be a bicategory. A morphism in $\mathcal{K}$ is a* map *or is* representable *if it admits a right adjoint.*

The piece of terminology *representable* is adapted from Hermida'a work involving bimodules (*cf.* [14, p.8]).

**Proposition 2.2** *A morphism in* **Span** *is representable precisely when its left leg is iso.*

**Corollary 2.3** *The isomorphism class of a representable span has a span with identity left leg as a canonical representant.*

**Definition 2.4** *Let $s, t : \mathcal{K} \to \mathcal{L}$ be lax functors. A* laxrep *transform $\alpha : s \Rightarrow t$ is a lax natural transformation with representable components at objects.*

Special cases of interest here are laxrep transforms among lax functors from a category to **Span**. Let $p, q : \mathbb{B} \to$ **Span** be such lax functors and let $\alpha : p \Rightarrow q$ be a laxrep transform. Its data w.r.t. to $\mathbb{B} \ni f : x \to y$ is given by the lax square

[2] Of the basic kind, we do not address issues like *exceptions* here.

$$\begin{tikzcd}
p(x) & p(x) \arrow[l, "\mathbf{id}"'] \arrow[r, "\alpha_x"] & q(x) \\
p(f) \arrow[u, "p(f)_1"] \arrow[d, "p(f)_2"'] & {} \arrow[phantom, "\alpha_f"] & q(f) \arrow[u, "q(f)_1"'] \arrow[d, "q(f)_2"] \\
p(y) & p(y) \arrow[l, "\mathbf{id}"] \arrow[r, "\alpha_y"'] & q(y)
\end{tikzcd}$$

where $\alpha_f$ is a morphism of spans

$$\begin{tikzcd}
 & p(f) \arrow[dl, "p(f)_1"'] \arrow[dr, "\alpha_y \circ p(f)_2"] \arrow[dd, "\alpha_f"] & \\
p(x) & & q(y) \\
 & p(x) \times_{q(x)} q(f) \arrow[ul, "p_1"] \arrow[ur, "q(f)_2 \circ p_2"'] &
\end{tikzcd}$$

**Definition 2.5 (Lax slice)** *Let $\mathcal{K}$ be a bicategory. The lax slice category $\mathsf{F} /\!/ \mathcal{K}$ is given by the data*

*(i) Objects: normalized pseudo-functors from a free category to $\mathcal{K}$*

*(ii) Morphisms: given $s : \mathsf{F}G \to \mathcal{K}$ and $t : \mathsf{F}H \to \mathcal{K}$ normalized pseudo-functors, a morphism $\alpha : s \to t$ is a laxrep transform $\alpha : s \Rightarrow t \circ \mathsf{F}k$ where $k : G \to H$ is a homomorphism of reflexive graphs*

*(iii) Composition: $\beta \circ \alpha = (l \circ k, l\beta \circ \alpha)$ where $\alpha : s \Rightarrow t \circ \mathsf{F}k$ and $\beta : t \Rightarrow u \circ \mathsf{F}l$ while the vertical composition $l\beta \circ \alpha$ is given by componentwise pasting*

We call objects of $\mathsf{F} /\!/$ **Span** *categorical transition systems* or *cts*'s so let **Cts** $\stackrel{def}{=}$ $\mathsf{F} /\!/$ **Span** from now on.

**Proposition 2.6** $\mathsf{F} /\!/$ **Span** $(\mathbb{B})$ *has finite limits provided* $\mathbb{B}$ *has.*

Proposition 2.6 is true for general reasons (*cf.* [12,20]). Finite limits in **Cts** admit a computational reading.

Consider the parallel composition without interaction

$$\texttt{x:=5}\ [x : nat] \quad \texttt{||} \quad \texttt{z:=7}\ [z : nat]$$

The cts

$$\begin{tikzcd}
\mathbb{N}^2 & \mathbb{N}^2 \arrow[l, "\mathbf{id}"'] \arrow[r, "\langle \mathbf{id},\, \lambda x{:}nat.7 \rangle"] & \mathbb{N}^2 \\
\mathbb{N}^2 \arrow[u, "\mathbf{id}"] \arrow[d, "\langle \lambda x{:}nat.5,\, \mathbf{id} \rangle"'] & \mathbb{N}^2 \arrow[ul, "\mathbf{id}"'] \arrow[dr, "\langle \lambda x{:}nat.5,\, \lambda x{:}nat.7 \rangle"'] & \mathbb{N}^2 \arrow[u, "\mathbf{id}"'] \arrow[d, "\langle \lambda x{:}nat.5,\, \mathbf{id} \rangle"] \\
\mathbb{N}^2 & \mathbb{N}^2 \arrow[l, "\mathbf{id}"] \arrow[r, "\langle \mathbf{id},\, \lambda x{:}nat.7 \rangle"'] & \mathbb{N}^2
\end{tikzcd}$$

corresponding to the intuition about this situation is a product object in **Cts**.

This example illustrates why control is represented by reflexive graphs in this setup: in order to take the interleavings of transitions induced by concurrent executions into account.

Consider the parallel composition with interaction

$$\texttt{x:=5;c!(x+x)}\ [x : nat] \quad \texttt{||} \quad \texttt{c?z;z:=z*z}\ [z : nat]$$

where the processes communicate over a typed channel $c : nat$ in CSP manner. The situation is reflected by the diagram

$$\begin{tikzcd}
\mathbb{N} & \mathbb{N} \arrow[l, "\mathbf{id}"'] \arrow[r, "!"] & \mathbf{id} & \mathbb{N} \arrow[l, "!"'] \arrow[r, "\mathbf{id}"] & \mathbb{N} \\
\mathbb{N} \arrow[u, "\mathbf{id}"] \arrow[d, "\lambda x:nat.5"'] \arrow[rr, phantom, "\alpha_v \Nearrow"] & & \mathbf{id} \arrow[u] \arrow[d] \arrow[rr, phantom, "\beta_{\mathbf{id}} \Nwarrow"] & & \mathbb{N} \arrow[u, "\mathbf{id}"'] \arrow[d, "\mathbf{id}"'] \\
\mathbb{N} & \mathbb{N} \arrow[l, "\mathbf{id}"'] \arrow[r, "!"] & \mathbf{id} & \mathbb{N} \arrow[l, "!"'] \arrow[r, "\mathbf{id}"] & \mathbb{N} \\
\mathbb{N} \arrow[u, "\mathbf{id}"] \arrow[d, "\mathbf{id}"'] \arrow[rr, phantom, "\alpha_v \Nearrow"] & & \mathbb{N} \arrow[u, "!"] \arrow[d, "!"'] \arrow[rr, phantom, "\beta_m \Nwarrow"] & & \mathbb{N}^2 \arrow[u, "\pi_1"'] \arrow[d, "\pi_2"'] \\
\mathbb{N} & \mathbb{N} \arrow[l, "\mathbf{id}"'] \arrow[r, "!"] & \mathbf{id} & \mathbb{N} \arrow[l, "!"'] \arrow[r, "\mathbf{id}"] & \mathbb{N} \\
\mathbb{N} \arrow[u, "\mathbf{id}"] \arrow[d, "\mathbf{id}"'] \arrow[rr, phantom, "\alpha_{\mathbf{id}} \Nearrow"] & & \mathbf{id} \arrow[u] \arrow[d] \arrow[rr, phantom, "\beta_n \Nwarrow"] & & \mathbb{N} \arrow[u, "\mathbf{id}"'] \arrow[d, "\lambda z:nat.z*z"'] \\
\mathbb{N} & \mathbb{N} \arrow[l, "\mathbf{id}"'] \arrow[r, "!"] & \mathbf{id} & \mathbb{N} \arrow[l, "!"'] \arrow[r, "\mathbf{id}"] & \mathbb{N}
\end{tikzcd}$$

in **Cts** with $\alpha$'s and $\beta$'s non-obvious components given by (the morphisms of spans) $\alpha_v = \langle \mathbf{id}, \lambda x : nat.x + x\rangle$ and $\beta_m = \langle \pi_1, \pi_2\rangle = \mathbf{id}$. It is a diagram of shape $\bullet \to \bullet \leftarrow \bullet$ where the object in the middle represents the communication channel. A pullback object stemming from this diagram is

$$\begin{tikzcd}
\mathbb{N}^2 \\
\mathbb{N}^2 \arrow[u, "\mathbf{id}"'] \arrow[d, "<(\lambda x:nat.5) \circ \pi_1{,}\pi_2>"'] \\
\mathbb{N}^2 \\
\{(a, (b, a + a)) \mid a, b \in \mathbb{N}\} \arrow[u, "<\pi_1{,}\pi_1 \circ \pi_2>"'] \arrow[d, "<\pi_1{,}\pi_2 \circ \pi_2>"'] \\
\mathbb{N}^2 \\
\mathbb{N}^2 \arrow[u, "\mathbf{id}"'] \arrow[d, "<\pi_1{,}(\lambda z:nat.z*z) \circ \pi_2>"'] \\
\mathbb{N}^2
\end{tikzcd}$$

again corresponding to the intuition about the situation at hand. Observe that spans which are not relations are involved here.

The examples suggest a notion of common behavior of interacting processes *qua limit* (*cf.* [11]) and a semantics of *remote procedure calls* or of *concurrent objects* can be constructed along these lines (*cf.* [21]). What is the relationship of a cts representing a program to an operational semantics of the latter, if any? A classic result shows the way.

**Definition 2.7** *A functor $s : \mathbb{B} \to \mathbb{C}$ has the* unique lifting of factorizations *or* ulf *property if, given $u \in \mathbb{B}$ and $\mathbb{C} \ni f = h \circ g$ s.t. $s(u) = f$, there are unique $v, w \in \mathbb{B}$ s.t. $u = w \circ v$ with $s(v) = g$ and $s(w) = h$.*

Let $\mathbf{Ulf}/\mathbb{B}$ be the full subcategory of the slice category $\mathbf{Cat}/\mathbb{B}$ where the objects are ulf functors. Let $Psd_{lax,rep}[\mathbb{B}, \mathbf{Span}]$ be the category of normalized pseudo-functors from $\mathbb{B}$ to **Span** and laxrep transformations. There is the equivalence of categories

$$\mathbf{Ulf}/\mathbb{B} \simeq Psd_{lax,rep}[\mathbb{B}, \mathbf{Span}]$$

which is the discrete case of a broader equivalence discovered independently by Giraud and by Conduché in the early 70's (*cf.* [15]).

Let $\mathbf{Ulf}_{\mathsf{F}}^{\rightarrow}$ be the full subcategory of the comma-category $\mathbf{id_{Cat}} \downarrow \mathsf{F}$ where the objects are ulf functors. A variation of the classic equivalence above relevant to the present setup is

**Theorem 2.8** *There is an equivalence of categories*

$$\mathbf{Ulf}_{\mathsf{F}}^{\rightarrow} \simeq \mathbf{Cts}$$

**Proof.** The comprehended version of a cts $s$ *i.e.* its ulf counterpart $\pi_s : \overline{s} \to \mathsf{F}G$ is obtained by an appropriate Grothendieck construction: a morphism in $\overline{s}$ is of the form $(k, f) : (a, x) \to (b, y)$ with $a \in s(x)$ and $b \in t(x)$ while $k \in s(f)$ s.t. $k$ it is mapped on $a$ resp. $b$ by $s(f)$'s left resp. right leg. Further, we have

$$\begin{tikzcd}
\bar{s} \arrow[rr, "\bar{\alpha}"] \arrow[d, "\pi_s"'] \arrow[drr, phantom, "(*)"] & & \bar{t} \arrow[d, "\pi_t"] \\
\mathsf{F}G \arrow[rr, "\mathsf{F}h"] \arrow[dr, "s"'] \arrow[r, phantom, shift right=4, "\overset{\alpha}{\Rightarrow}"] & & \mathsf{F}H \arrow[dl, "t"] \\
& \mathbf{Span} &
\end{tikzcd}$$

where

$$\bar{\alpha} : \begin{array}{c}(a,x)\\ \Big\downarrow{\scriptstyle (k,f)}\\ (b,y)\end{array} \quad \mapsto \quad \begin{array}{c}(\alpha_x(a), \mathsf{F}h(x))\\ \Big\downarrow{\scriptstyle ((p_2 \circ \alpha_f)(k), \mathsf{F}h(f))}\\ (\alpha_y(b), \mathsf{F}h(y))\end{array}$$

The other way round it is enough to take the fibers. Let $s : \mathbb{D} \to \mathsf{F}G$ be an ulf functor and let $s_x$ resp. $s_f$ the set of objects over $x \in (\mathsf{F}G)_0$ resp. over $f \in (\mathsf{F}G)_1$. The functor $s$ determines a cts $\underline{s} : \mathsf{F}G \to \mathbf{Span}$ given by

$$\underline{s} : \begin{array}{c} x \\ \Big\downarrow f \\ y \end{array} \quad \mapsto \quad \begin{array}{c} s_x \\ \Big\uparrow dom \\ s_f \\ \Big\downarrow cod \\ s_y \end{array}$$

Let $(m, n) : s \to t$ be a morphism in $\mathbf{Ulf}_{\mathsf{F}}^{\to}$. It determines a laxrep transform $\underline{m} : \underline{s} \Rightarrow n \circ \underline{t}$ given at $f$ by $\underline{m}_x \stackrel{def}{=} m\,|_{s_x}$ and $\underline{m}_f\,(p) \stackrel{def}{=} (dom\,p, m\,(p))$. □

**Corollary 2.9** *The square* $(*)$ *is a pullback square provided* $\alpha = \mathbf{id}$.

In particular, when a cts stems from an imperative program, it can be shown that its comprehended version is essentially the projection from the *state-space* induced by an *early* operational semantics of the program onto the control flow graph of the latter (*cf.* [21]).

We borrow here a piece of terminology from a setting designed for categorical process algebras by Fiore (*cf.* [10]) and call $\overline{s}$ the *category of evolutions* corresponding to $s$. The codomain of an ulf functor is under the computational reading put forward in *op.cit.* a category *controlling* the evolutions. By analogy, we call $s$ a *control program.*

## 3 Path Simulation and Open Maps

Back to the classical setting of labeled transition systems, it can be argued that the *dynamics* of an lts is observed by composing transitions and recording the induced sequences of actions (*cf.* [9]). Given a presentation of a transition system as a homomorphism from the underlying graph of transitions to the one-vertex graph with individual labels as self-loops, the dynamics comes thus about *via* the free functor $\mathsf{F} \vdash \mathsf{U} : \mathbf{RGraph} \to \mathbf{Cat}$. It can be argued that most notions of simulation are inherently tied to *dynamics* in the sense of being defined in terms of certain paths, that is with dynamics being part of the domain of discourse.

This does not immediately meet the eye when considering strong simulation. Let namely $G$ be a graph, $\Sigma$ a set and $\Sigma^*$ the free monoid over $\Sigma$. A labeled transition system can also be presented as a homomorphism $t : G \to \Sigma^*$ to the graph underlying $\Sigma^*$ (the latter seen as a category) s.t. $G$'s individual transitions in are mapped on the generators. In this case, the formulation of strong simulation is the same in terms of statics and in terms of dynamics. However, already the classic notion of weak simulation can be presented in terms of dynamics: if we allow labels to take as value the empty sequence $\varepsilon$ as well, then the static and dynamic notions of simulation diverge and the latter corresponds to weak simulation. Clearly, the trick implicitly reintroduces the saturation monad, as seen at the example of $x \xrightarrow{a} y$ being simulated by $u \xrightarrow{\varepsilon} v \xrightarrow{a} w$.

The above considerations may be interesting since invisible actions are *really* invisible but represent in itself little more than yet another presentation of classic weak simulation. In this section, we elaborate on *path simulation* among cts's where the principle sketched above is applied systematically in that a transition can be matched by an arbitrary path as long as the labels agree, recall that cts's are labeled in an arbitrary bicategory so the composition of labels is not in general a mere concatenation of letters. We then proceed to give a characterization of path simulation in terms of an appropriate notion of *open maps* (*cf.* [2]).

**Definition 3.1** *Let* $s : \mathsf{F}G \to \mathbf{Span}$ *and* $t : \mathsf{F}H \to \mathbf{Span}$ *be cts's. A* path simulation *from* $s$ *to* $t$ *is a relation* $r \subseteq G_0 \times H_0$ *s.t.*

*(i)* $\iota_S\ (r)\ \iota_T$

*(ii)* $x\ (r)\ x'\ \&\ x \xrightarrow{f} y \in \mathsf{F}G \ \Rightarrow\ \exists\, x' \xrightarrow{f'} y' \in \mathsf{F}H.\, y\ (r)\ y' \quad \& \quad s\,(f) = t\,(f')$

Consider the programs

$$\texttt{x:=7}\ [x : nat]$$

and

$$\texttt{x:=5;x:=x+2}\ [x : nat]$$

along with the corresponding cts's

$$\mathbb{N} \xleftarrow{\mathbf{id}} \mathbb{N} \xrightarrow{\lambda x.7} \mathbb{N}$$

and

$$\mathbb{N} \xleftarrow{\mathbf{id}} \mathbb{N} \xrightarrow{\lambda x.5} \mathbb{N} \xleftarrow{\mathbf{id}} \mathbb{N} \xrightarrow{\lambda x.x+2} \mathbb{N}$$

There is the obvious path simulation from the first cts to the second. Notice however that there would be no simulation at all if we had more conservatively rephrased strong simulation in terms of individual transitions exclusively. Such observations were undeniably the motivating factor for the introduction and study of path simulation. However, in order to be useful in the context of concurrency, it is necessary to *classify* the transitions in addition to the computations they carry since the latter are not *intrinsically* silent or observable. Indeed, as noticed by Lynch and Tuttle, if a transition simulates some other one then both need to be of the same *kind* in addition to carry the same computation (*cf.* [17]) and off course the same is true for control paths. A notion of $\chi$-simulation adapted from Cockett and Spooner's work on *process categories* (*cf.* [7]) provides an elegant classifying device (*cf.* [22]).

It is customary to characterize (bi)simulations in terms of appropriate notions of open maps (*cf.* [2]). The notion of opens maps for presheaves stating that every naturality square is a *quasi-pullback* (*cf.* [1]) turns out to be too strong here. Better adapted for the present setup is

**Definition 3.2** *A functor* $k : \mathbb{B} \to \mathbb{C}$ *is open in* **Cat** *provided any* $\mathbb{B} \ni f : t\,(a) \to c$ *lifts under* $k$. *The morphism of cts's* $\alpha : s \to t$ *where* $\alpha : s \Rightarrow t \circ \mathsf{F}h$ *is* open *provided* $\alpha = \mathbf{id}$ *and* $\mathsf{F}h$ *is open in* **Cat**.

and path simulation can be characterized as follows

**Theorem 3.3** *Let $s : \mathsf{F}G \to \mathbf{Span}$ and $t : \mathsf{F}H \to \mathbf{Span}$ be cts's. There is a path simulation from $s$ to $t$ iff there is a graph $R$, a surjective graph homomorphism $d : R \to G$ and a functor $C : \mathsf{F}(R) \to \mathsf{F}(H)$ s.t.*

$$\begin{tikzcd}
& \mathsf{F}R \arrow[dl, "\mathsf{F}d"'] \arrow[dr, "C"] & \\
\mathsf{F}G \arrow[dr, "s"'] & & \mathsf{F}H \arrow[dl, "t"] \\
& \mathbf{Span} &
\end{tikzcd}$$

*commutes and $\mathsf{F}d$ is open.*

Definition 3.2 and theorem 3.3 represent a variation on the well-known theme of open maps. The computational intuition behind the setup is of $d$ *sampling* $s$'s calculations and matching them along $t$'s control paths.

**Proposition 3.4** *Isomorphisms are open maps. Open maps are composition- and pullback-stable.*

What is the relation between path simulations on control programs and on the corresponding categories of evolutions? Suppose cts $t$ (path) simulates cts $s$. We then have

$$\begin{tikzcd}
& \bar{r} \arrow[dl, "\bar{d}"'] \arrow[dr, "\bar{c}"] \arrow[dd, "\pi_r"] & \\
\bar{s} \arrow[dd, "\pi_s"'] \arrow[r, phantom, "(**)"] & & \bar{t} \arrow[dd, "\pi_t"] \\
& \mathsf{F}r \arrow[dl, "\mathsf{F}d"] \arrow[dr, "c"'] & \\
\mathsf{F}G \arrow[dr, "s"] & & \mathsf{F}H \arrow[dl, "t"'] \\
& \mathbf{Span} &
\end{tikzcd}$$

with $\mathsf{F}d$ an open map. It is further the case by corollary 2.9 that square $(**)$ in the diagram above is a pullback square. Hence, by proposition 3.4, $\overline{d}$ is open too so in particular we have a path simulation from $\overline{s}$ to $\overline{t}$. Alternatively, we can start from the ulf functors as processes *i.e.* terms of a canonical process algebra (*cf.* [10]). Then the pair $\left(\overline{d}, \mathsf{F}(d)\right)$ can be seen as an open map from $\pi_s$ to $\pi_t$ under this computational interpretation.

## 4 A Relational Structure

As noticed in section 3, characterizing (bi)simulations in terms of an appropriate notion of open maps has become customary. Our definition 3.2 of open maps is of direct use for a proof of theorem 3.3 but does not disclose any computational intuition. The latter would be that an open map has a *unique lifting property* w.r.t. to paths chosen in a *subcategory of paths* $\mathcal{P} \subseteq \mathbf{Cts}$: it is the celebrated weak orthogonality property

$$\begin{tikzcd} P \arrow[r, "u"] \arrow[d, "p"'] & Q \arrow[d, "q"] \arrow[dl, dotted] \\ X \arrow[r, "f"'] & Y \end{tikzcd}$$

with $P$ and $Q$ path objects, at heart of such considerations. Typical investigations are based on a *model of concurrency* $\mathcal{C}$ *e.g.* the category of lts's and their morphisms, and on a distinguished category of paths $\mathcal{P} \subseteq \mathcal{C}$ therein. Fixing $\mathcal{C}$ and variating $(\mathcal{P})$ characterizes different notions of (bi)simulation w.r.t. $\mathcal{C}$. Notice that this indirectly substantiates the claim made in section 3 that simulations are about dynamics.

Observe however that the approach has the defect that $\mathcal{P}$ is subject to an *arbitrary choice*: nothing prescribes *what* kind of paths do live in $\mathcal{P}$. Put differently, the choice of $\mathcal{P}$ is justified *a posteriori* in concrete instances. In this section, we investigate the algebraic structure of path simulation from a different angle, attempting to avoid the defect mentioned above. Our model of concurrency being general enough, we investigate if there is an *a priori* structure characterizing path simulations. A further classic construction gives a positive answer to the question: we identify Bénabou's *bicategory of cylinders* (*cf.* [4]) as the seeked *relational structure*.

This section builds on issues originally considered by Hermida in the context of lts's (*cf.* [13]). Given a alphabet $\Sigma$, let $S = (S, \rightarrow \subseteq S \times \Sigma \times S)$ be an lts over $L$ and notice that it can be presented as as an indexed family $(\rightarrow_\alpha \subseteq S \times S)_{\alpha \in \Sigma}$ . Let $S' = (\rightarrow'_\alpha \subseteq S' \times S')_{\alpha \in \Sigma}$ be a further lts over $\Sigma$. Hermida's key observation is that $r \subseteq S \times S'$ is a simulation relation precisely when

$$\forall \alpha \in \Sigma.\ (\rightarrow_\alpha \circ r^{op}) \subseteq (r^{op} \circ \rightarrow'_\alpha)$$

which directly leads to the construction of a bicategory of cylinders. The message is similar here but the technical details are more complex since all paths witnessing a simulation need to be accounted for.

**Definition 4.1** *A quantale $Q$ is a lattice with joins and bottom, equipped with a tensor $\otimes : Q \times Q \to Q$ with unit $1_Q$, distributing over the joins and s.t. $\perp \otimes z = z \otimes \perp = \perp$. The category $\mathbb{Q}$ is given by the data*

- *Objects: quantales*
- *Morphisms: monoidal functors preserving the joins and bottom*

**Proposition 4.2** *A category $\mathbf{C}$ determines a quantale $\mathcal{Q}(\mathbf{C})$ with underlying lattice $\mathcal{P}(\mathbf{C}_1)$ and tensor given by*

$$A \otimes B \stackrel{def}{=} \{g \circ f \mid (f, g) \in A \times_{C_1} B\}$$

*In particular $1_{\mathcal{Q}(\mathbf{C})} = \{\mathbf{id}_X \mid X \in \mathbf{C}\}$. The assignment extends pointwise to a* *functor* $\mathcal{Q}(\_) : \mathbf{Cat} \to \mathbb{Q}$.

**Definition 4.3** *The 2-category* $\mathbb{P}$ *is given by the data*

*(i)* *Objects: pairs* $(E, A)$ *for* $E \in \mathbb{Q}$ *and* $A$ *a set*

*(ii)* *Morphisms: pairs* $(f, \alpha) : (E, A) \to (F, B)$ *for* $E \xrightarrow{f} F \in \mathbb{Q}$ *and*

$$\alpha : A \times B \to F$$

*a function to* $F$*'s underlying set*

*(iii)* *2-cells: pointwise order*

*(iv)* *Composition:* $(g, \beta) \circ (f, \alpha) \stackrel{def}{=} (g \circ f, \beta \odot_g \alpha)$ *where*

$$(\beta \odot_g \alpha)(a, c) \stackrel{def}{=} \bigvee_{b \in B} ((g \circ \alpha)(a, b) \otimes \beta(b, c))$$

*(v)* *Identities:* $\mathbf{id}_{(E,A)} \stackrel{def}{=} (\mathbf{id}_E, \delta_{A,E})$ *where*

$$\begin{aligned} \delta_{A,E} : A \times A &\to E \\ (a, a') &\mapsto \begin{cases} 1_E & a = a' \\ \perp \end{cases} \end{aligned}$$

Observe that $\mathbb{P}$ is 2-cofibred over $\mathbb{Q}$. Although the bicategorical infrastructure is indeed very simple, the setup as a whole is as we will see of a rather bicategorical nature, so we keep the terminology. An endomorphism with identity in its first component is called *normalized*. $\mathbb{P}$'s morphisms in are essentially quantale-valued relations, in particular

**Definition 4.4** *A morphism* $(f, \alpha) : (E, V) \to (F, W)$ *in* $\mathbb{P}$ *is a* $\mathbb{P}$*-relation if*

$$\alpha(v, w) \in \{\perp_F, 1_F\}$$

*for all* $v \in V$ *and* $w \in W$*. It is an* $\mathbb{P}$*-map if it is a* $\mathbb{P}$*-relation s.t.*

$$\forall (v \in V), (w, w' \in W).\ (\alpha(v, w) = 1_F \quad \& \quad \alpha(v, w') = 1_F) \Rightarrow w = w'$$

We next introduce the notion of *premodule* central to this characterization of path simulation:

**Definition 4.5** *Let* $\mathbb{L}$ *be a category and* $\mathbb{M}$ *a bicategory. An* $\mathbb{L}$-premodule *in* $\mathbb{M}$ *is an object* $M \in \mathbb{M}$ *along with a lax functor* $L : \mathbb{L} \to \mathbb{M}$ *s.t.* $LX = M$ *for all* $X \in \mathbb{L}$*.* $L$ *is called the premodule's* action*. A premodule's action is* normalized *if it is the case pointwise.* $\mathbb{L}$*-premodules in* $\mathbb{M}$ *are organized in the category* $\mathbf{Premod}_{\mathbb{L},\mathbb{M}}$ *together with lax transformations among their actions.*

Traditionally, a module is an abelian group $G$ equipped with a ring action $R \times G \to G$ (*cf.* for instance [16]). Fixing an element of $R$ induces an endomorphism of the group. We consider here a generalization of this view where an object of a bicategory plays the rôle of the group. In particular, the premodules of definition 4.5 have nothing in common with the well-known notion variously called *(bi)module*, *profunctor* or *distributor* (*cf.* [4]). We choose the term premodule since we feel that calling something a *module* would be justified in presence of structures being or generalizing at least additive categories. In this context, the choice of the name only refers to lax functors constant on objects.

For the sake of conciseness, we consider in the following a simplified version of the notion of cts's and their morphisms.

**Definition 4.6** *Let* $\mathbb{L}$ *be a category.* Strict categorical transition systems *or* scts's *over* $\mathbb{L}$ *are organized in the slice category* $\mathsf{F}/\mathbb{L} \overset{def}{=} \mathbf{Scts}$.

Let $\mathbf{Span}^\sharp$ be $\mathbf{Span}$'s classifying category. Any cts $\mathsf{F}G \to \mathbf{Span}$ gives rise to an scts $\mathsf{F}G \to \mathbf{Span}^\sharp$ and it is easily seen that the lax structure in $\mathbf{Cts}$ is in practice relevant mostly for calculating behaviors of networks of processes: the resulting cts can always be transformed in an scts. Given the obvious notion of path simulation among scts's, there is not much generality lost in simplifying the setup this way.

Let $\lfloor \mathbf{C} \rfloor \overset{def}{=} (\mathcal{Q}(\mathbf{C}), \mathbf{C}_0) \in \mathbb{P}$ for a category $\mathbf{C}$. Given a further category $\mathbb{D}$ and a functor $F : \mathbf{C} \to \mathbb{D}$, the morphism

$$\mathbb{P} \ni \lfloor F \rfloor \overset{def}{=} \left(\mathcal{Q}(F), \overline{F_0}\right) : \lfloor \mathbf{C} \rfloor \to \lfloor \mathbb{D} \rfloor$$

with $\overline{F_0}$ the $\mathbb{P}$-map induced by $F_0$.

**Theorem 4.7** *There is a functor*

$$(\_)^\sharp : \mathbf{Scts} \to \mathbf{Premod}_{\mathbb{L},\mathbb{P}}$$

**Proof.**Let $s : \mathsf{F}G \to \mathbb{L}$ be an scts, $x \xrightarrow{f} y \in \mathbb{L}$ and $\mathsf{F}G_f(a,b) \overset{def}{=} s^{-1}(f) \cap \mathsf{F}G(a,b)$. $\lfloor \mathsf{F}G \rfloor \in \mathbb{P}$ is a premodule with normalized action $s^\sharp : \mathbb{L} \to \mathbb{P}$ given by the assignment

$$f \mapsto \left(\mathbf{id}_{\mathcal{Q}(\mathsf{F}G)},\ \lambda(a,b) \,.\, \mathsf{F}G_f(a,b)\right)$$

Let $t : \mathsf{F}H \to \mathbb{L}$ be a further scts and let $h : G \to H$ be a graph homomorphism giving rise to a morphism of scts's. The arrow part of the assignment is then $(\mathsf{F}h)^\sharp \overset{def}{=} \lfloor \mathsf{F}h \rfloor : s^\sharp \Rightarrow t^\sharp$ □

**Definition 4.8** *The* bicategory of cylinders $\mathcal{C}yl$ *is given by the data*

*(i) Objects: morphisms* $(f,\alpha) : (E,V) \to (F,W)$ *in* $\mathbb{P}$

*(ii) Morphisms: a morphism* $(f,\alpha) \to (f',\alpha')$ *is a pair* $((g,\beta),(h,\gamma))$ *giving rise to an oplax square*

$$\begin{tikzcd}
(E,V) \arrow[r, "(f,\alpha)"] \arrow[d, "(g,\beta)"'] \arrow[dr, phantom, "\supseteq"] & (F,W) \arrow[d, "(h,\gamma)"] \\
(E',V') \arrow[r, "(f',\alpha')"'] & (F',W')
\end{tikzcd}$$

*in* $\mathbb{P}$

*(iii) 2-cells, composition and units: pointwise in* $\mathbb{P}$

*The homomorphism of bicategories* $\Re : \mathcal{C}yl \to \mathbb{P} \times \mathbb{P}$ *acts as* $\langle dom, cod \rangle$ *on objects and as identity on morphisms and 2-cells.*

$\mathcal{C}yl$ is a bicategory derived from $\mathbb{P}$ by means of a generalized Grothendieck construction applied to the "hom-functor" $\mathbb{P}^{op} \times \mathbb{P} \to \mathbf{Cat}$ (*cf.* [19]).

**Theorem 4.9** *Let* $s : G \to \mathbb{L}$ *and* $t : H \to \mathbb{L}$ *be scts's. Are equivalent*

*(i) there is a path simulation from* $s$ *to* $t$

*(ii) the triangle*

$$\begin{tikzcd}
 & \mathcal{C}yl \arrow[d, "\Re"] \\
\mathbb{L} \arrow[ur, "\sigma"] \arrow[r, "<t^\sharp, s^\sharp>"'] & \mathbb{P} \times \mathbb{P}
\end{tikzcd}$$

*commutes for a premodule* $\sigma$

**Proof.**"$\Rightarrow$" Let $r \subseteq G_0 \times H_0$ be a path simulation from $s$ to $t$. The $\mathbb{P}$-relation

$$(p, \overline{r^{op}}) : \lfloor \mathsf{F}H \rfloor \to \lfloor \mathsf{F}G \rfloor$$

given by

$$p : \mathcal{Q}(\mathsf{F}H) \to \mathcal{Q}(\mathsf{F}G)$$
$$k \mapsto \begin{cases} \mathcal{Q}(\mathsf{F}G) & \text{if } k \notin \{\perp, 1\} \\ 1 & \text{if } k = 1 \\ \perp & \text{otherwise} \end{cases}$$

and

$$r^{op} : H_0 \times G_0 \to \mathcal{Q}(\mathsf{F}G)$$
$$(m, b) \mapsto \begin{cases} 1 & \text{if } b\ (r)\ m \\ \perp & \text{otherwise} \end{cases}$$

is an $\mathbb{L}$-premodule in $\mathcal{C}yl$ with action given by the assignment

$$f \mapsto \left(t^\sharp f,\ s^\sharp f\right)$$

"$\Leftarrow$" Let $(t, \rho) : \lfloor \mathsf{F}H \rfloor \to \lfloor \mathsf{F}G \rfloor$ be a premodule in $\mathcal{C}yl$ with action $\sigma : \mathbb{L} \to \mathcal{C}yl$ s.t. $\Re \circ \sigma = \left\langle t^\sharp, s^\sharp \right\rangle$. The relation $r \subseteq G_0 \times H_0$ given by

$$b\ (r)\ m \overset{def}{\Leftrightarrow} \rho\,(m, b) = 1_{\mathcal{Q}(\mathsf{F}H)}$$

is a path simulation. □

Following Hermida (*op.cit.*), path simulations can be seen as binary predicates over the involved scts's. In particular, the present setup exhibits a clear distinction as to *where* do scts's resp. path simulations live. The distinction reflects the fact that a simulation is a *logical* notion while scts-like entities belong to the *effective* world. This is to be contrasted with the approach using open maps.

## 5 Concluding Remarks

We have identified the *lax slice* $\mathsf{F}/\!/\mathbf{Span}\,(\mathbb{B})$ as a category of *models of concurrency* called *categorical transition systems* and demonstrated their relevance in giving meaning to a range of everyday phenomena including message passing among imperative programs. We also exhibited the relationship to operational semantics via the the classical equivalence $\mathbf{Ulf}/\mathbb{C} \simeq Psd_{lax.rep}\,[\mathbb{C}, \mathbf{Span}]$. We further identified the bicategory of spans $\mathbf{Span}\mathsf{F} /\!/ \mathbf{Span}\,(\mathbb{B})$ as organizing *processes* at a basic level and addressed the question of simulation in this context.

We have argued that simulation should be studied as a run-time or *dynamic* phenomenon and formalized the notion as *path simulation.* and provided two abstract characterizations of path simulation: in terms of open maps and in terms of a relational structure *qua* Bénabou's bicategory of cylinders.

### 5.1 Related Work

The approach making use of open maps is documented by a large body of work characterizing important variants of (bi)simulations (*cf.* [5] for a systematic study) and investigating conditions of functorial stability (*cf.* [6]). Section 4 builds on and expands topics originally considered by Claudio Hermida (*cf.* [13]).

Lindsay Errington introduced a much more general notion of categorical transition systems in his doctoral thesis [8] using a presentation he calls *twisted systems.* Given a category of *computational shapes* $\mathbf{Shp}$ and a functor $\kappa : \mathbf{Shp} \to \mathbf{Cat}$, Errington's cts's are pairs $\left(J,\, \kappa J \xrightarrow{S} C\right)$ where $J \in \mathbf{Shp}$ and $S$ is a functor from $\kappa J$ to a category $C$. His notion of *bisimulation* is expressed in terms of statics and characterized using the technique of *open maps.*

Marcelo Fiore considers in [10] a broad notion of processes embodied by *ulf functors.* It turns out that ulf functors cover a broad class of processes ranging from discrete to continuous systems. Fiore's notion of bisimulation is technically like ours and Errington's. He characterizes *bisimulation* abstractly in terms of *canonically* given open maps.

### *5.2 Future Research*

The notion of path simulation is still fairly crude w.r.t. computations in that it discriminates on-the-nose, *i.e.* a simulating control path needs to carry the *same* computation than the simulated one. Generalizing the setup will give rise to a notion of *lax path simulation* where the discriminating criterion will be a 2-cell, paving the way to a setting relevant for applications like program development by *refinement of specifications*.

The virtue of the approach in section 4 is at the same time a defect. Since $(\_)^\sharp$ squeezes everything into one object, we were not able to exhibit an equivalence of categories among scts and premodules. In fact, a more general approach making use of distributors seems promising, also because a link to categorical modal logic appears possible.

## Acknowledgments

I am deeply indebted to Claudio Hermida for his suggestions, criticism and the many common hours spent working in diverse urban environments. I am very indebted to Robin Cockett for the many suggestions and discussions, the canadian winter at it's best. Thanks to prof. Petitpierre for his encouragements and support. I am indebted to Christoph Sprenger and to Lindsay Errington for their kindness and their constructive criticism, in particular to Christoph for his careful proof reading of a previous version of this manuscript. I would like to thank the anonymous referees for their pertinent remarks.

## References

[1] A.Joyal and I.Moerdijk. *Algebraic Set Theory*, volume 220 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 1995.

[2] A.Joyal, M.Nielsen, and G.Winskel. Bisimulation and open maps. *Information and Computation*, 1996.

[3] Michael Barr and Charles Wells. *Toposes, Triples and Theories*. Springer Verlag, Berlin, 1984.

[4] Jean Bénabou. *Introduction to Bicategories*, volume 47 of *Lecture Notes in Mathematics*. Springer, 1967.

[5] A. Cheng and M. Nielsen. Open maps at work, 1995.

[6] Allan Cheng and Mogens Nielsen. Open maps, behavioural equivalences, and congruences. In *Colloquium on Trees in Algebra and Programming*, pages 257–271, 1996.

[7] J. R. B. Cockett and D. A. Spooner. Constructing process categories. *Theoretical Computer Science*, 177(1), April 1997.

[8] Lindsay Erington. *Twisted Systems*. PhD thesis, Department of Computing, Imperial College, London, 1999.

[9] Gianluigi Ferrari, Ugo Montanari, and Miranda Mowbray. Structured transition systems with parametric observations: observational congruences and minimal realizations. *Mathematical Structures in Computer Science*, 7(3):241–282, June 1997.

[10] Marcelo Fiore. Fibred models of processes: Discrete, continuous, and hybrid systems. In *Proceedings of the IFIP TCS2000 Conference*, pages 457–473. Springer-Verlag, 2000.

[11] J Goguen. A categorial manifesto. Technical Report PRG-72, Programming Research Group, University of Oxford, March 1989.

[12] J. W. Gray. *Proceedings of the Conference on Categorical Algebra, La Jolla*, chapter Fibered and Cofibered Categories, pages 21–83. Springer, 1966.

[13] Claudio Hermida. Simulations as modules. Unpublished, 1997.

[14] Claudio Hermida. From coherent structures to universal properties. *Journal of Pure and Applied Algebra*, 165(1):7–61, 2001.

[15] Peter T. Johnstone. A note on discrete conduché fibrations. *Theory and Applications of Categories*, 5(1):1–11, 1999.

[16] Serge Lang. *Algebra*. Adison-Wesley, 1984.

[17] Nancy A. Lynch and Mark R. Tuttle. An introduction to Input/Output automata. *CWI-Quarterly*, 2(3):219–246, September 1989.

[18] R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer-Verlag, 1980.

[19] Ross Street. Fibrations in bicategories. *Cahiers de Topologie et Géométrie Différentielle*, 21(2), 1980.

[20] A. Tarlecki, R. M. Burstall, and J. A. Goguen. Some fundamental algebraic tools for the semantics of computation part 3: Indexed categories. *preprint*, 1989.

[21] Krzysztof Worytkiewicz. *Components and Synchronous Communication in Categories of Processes*. PhD thesis, Swiss Federal Institute of Technology, Lausanne, 2000.

[22] Krzysztof Worytkiewicz. Concrete process categories. To appear in the proceeding of CMICM, August 2002.